



(11) **EP 3 072 134 B1**

(12) **EUROPEAN PATENT SPECIFICATION**

(45) Date of publication and mention
of the grant of the patent:
04.10.2023 Bulletin 2023/40

(51) International Patent Classification (IPC):
G11C 29/00 ^(2006.01) **G11C 16/06** ^(2006.01)
G06F 11/10 ^(2006.01)

(21) Application number: **14864526.0**

(52) Cooperative Patent Classification (CPC):
G06F 11/0751; G06F 11/073; G06F 11/1012

(22) Date of filing: **21.10.2014**

(86) International application number:
PCT/US2014/061486

(87) International publication number:
WO 2015/076963 (28.05.2015 Gazette 2015/21)

(54) **DEFECT MANAGEMENT POLICIES FOR NAND FLASH MEMORY**

DEFEKTVERWALTUNGSRICHTLINIEN FÜR NAND-FLASH-SPEICHER

RÈGLES DE GESTION DE DÉFAUTS POUR MÉMOIRE FLASH NAND

(84) Designated Contracting States:
**AL AT BE BG CH CY CZ DE DK EE ES FI FR GB
GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO
PL PT RO RS SE SI SK SM TR**

- **RAGHUNATHAN, Shyam Sunder**
Sunnyvale, California 94087 (US)
- **MOTWANI, Ravi H.**
San Diego, California 92128 (US)

(30) Priority: **22.11.2013 US 201314087282**

(74) Representative: **Rummler, Felix**
Maucher Jenkins
Liebigstrasse 39
80538 München (DE)

(43) Date of publication of application:
28.09.2016 Bulletin 2016/39

(73) Proprietor: **Intel Corporation**
Santa Clara, CA 95054 (US)

(56) References cited:
WO-A2-2012/166726 US-A1- 2009 044 085
US-A1- 2010 115 376 US-A1- 2010 122 148
US-A1- 2010 122 148 US-A1- 2010 185 906
US-A1- 2011 055 468 US-A1- 2012 159 285

(72) Inventors:

- **KALAVADE, Pranav**
San Jose, California 95129 (US)
- **ZHU, Feng**
San Jose, California 95124 (US)

Note: Within nine months of the publication of the mention of the grant of the European patent in the European Patent Bulletin, any person may give notice to the European Patent Office of opposition to that patent, in accordance with the Implementing Regulations. Notice of opposition shall not be deemed to have been filed until the opposition fee has been paid. (Art. 99(1) European Patent Convention).

EP 3 072 134 B1

Description

TECHNICAL FIELD

[0001] Embodiments described herein relate generally to nonvolatile memory technology, and more specifically to systems and methods of managing defects in nonvolatile memory.

BACKGROUND

[0002] In a conventional storage system, such as a nonvolatile storage system, data can be stored in accordance with a predetermined error correction coding scheme, such as a low-density parity-check (LDPC) coding scheme. Such a nonvolatile storage system can include a plurality of nonvolatile memory (NVM) devices, each NVM device containing a multitude of NVM storage elements or cells for storing encoded data in the form of LDPC codewords. When reading LDPC codewords from such NVM devices, one or more reference threshold voltage levels can be established between a plurality of programming states of each NVM cell, and a threshold voltage level of the NVM cell can be compared to the respective reference threshold voltage levels to determine the actual programming state of the NVM cell.

[0003] In a typical mode of operation, in order to read a desired LDPC codeword from a selected NVM device, a hard bit read operation can be performed, in which the LDPC codeword is read from the selected NVM device as so-called "hard data" (*i.e.*, the logical high and low bit levels in the LDPC codeword are taken as a "1" and "0", respectively). Further, a soft bit read operation can be performed, in which the LDPC codeword is read from the selected NVM device as so-called "soft data" (*i.e.*, the logical high and low bit levels in the LDPC codeword are taken as "1" and "0", respectively, and probability information is provided indicating the likelihood that a respective bit in the LDPC codeword is a "1" or "0"). The LDPC codeword read from the selected NVM device can then undergo error correction decoding to determine whether or not the LDPC codeword is valid. If the decoded LDPC codeword is found to be invalid or "faulty" (*e.g.*, there may have been an error correction coding/decoding failure), then additional LDPC codewords can be read from the other NVM devices, and XOR data recovery can be performed using the additional LDPC codewords in an attempt to recover the desired LDPC codeword.

[0004] Such a detection of a faulty LDPC codeword may indicate, among other things, that a physical memory page on the selected NVM device is being affected by an underlying hardware defect (*e.g.*, an open or short circuit), or that the selected NVM device has an intrinsically high residual bit error rate (RBER). In any case, it would be desirable to have a mechanism for managing defects in nonvolatile storage systems that can be used to avoid an inadvertent loss of data, while maintaining as much useful memory in the nonvolatile storage systems

as possible.

[0005] US 2012/159285 A1 relates to storage drives with LDPC encoded data. It provides read techniques for recovering an errantly read memory unit (*e.g.*, faulty LDPC codeword). It also discloses building of extra redundancy into the system by providing an additional memory module with XOR values of corresponding LDPC codewords from the other memory modules (*e.g.*, dies) making up the storage drive. This enables recovery of an errantly read codeword.

SUMMARY OF THE INVENTION

[0006] The present invention is defined by the features of the independent claims. Preferred embodiments thereof are defined by features of the dependent claims.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] The accompanying drawings, which are incorporated in and constitute part of this specification, illustrate one or more embodiments described herein, and, together with the Detailed Description, explain these embodiments. In the drawings:

FIG. 1 a is a block diagram of an exemplary nonvolatile storage system including a nonvolatile memory (NVM) defect management policy engine, in accordance with the present application;

FIG. 1b is a block diagram of exemplary NVM read logic included in the nonvolatile storage system of FIG. 1a;

FIG. 2 is a diagram of "voltage" versus "probability" that represents the decoding of a codeword read from an NVM device included in the nonvolatile storage system of FIG. 1a;

FIG. 3 is a flow diagram illustrating an exemplary method of operating the NVM defect management policy engine included in the nonvolatile storage system of FIG. 1a; and

FIG. 4 is a block diagram of an exemplary computer system in which the nonvolatile storage system of FIG. 1a may be employed.

DESCRIPTION OF EMBODIMENTS

[0008] Systems and methods of managing defects in nonvolatile storage systems are disclosed that can be used to avoid an inadvertent loss of data, while maintaining as much useful memory in the nonvolatile storage systems as possible. The disclosed systems and methods monitor a plurality of trigger events for detecting possible defects in one or more nonvolatile memory (NVM) devices included in the nonvolatile storage systems, and apply one or more defect management policies to the respective NVM devices based on the types of trigger events that resulted in detection of the possible defects. Such defect management policies can be used proac-

tively to retire memory in the nonvolatile storage systems with increased granularity, focusing the retirement of memory on regions of nonvolatile memory that are likely to contain a defect.

[0009] FIG. 1a depicts an illustrative embodiment of an exemplary system 100 for accessing one or more NVM devices in a nonvolatile storage system, in accordance with the present application. As shown in FIG. 1a, the system 100 includes a host processor 102, a nonvolatile storage system 104, and a memory bus 105 operative to communicably couple the host processor

102 to the nonvolatile storage system 104. The nonvolatile storage system 104 includes an NVM controller 106, one or more NVM devices 108.1-108.n communicably coupled to the NVM controller 106, and an NVM defect management policy engine 110 communicably coupled to the NVM controller 106. As further shown in FIG. 1a, the NVM controller 106 includes NVM read logic 112 for use in reading data from the respective NVM devices 108.1-108.n, as well as NVM write logic 114 for use in writing data to the respective NVM devices 108.1-108.n, in response to one or more data read/write requests issued by the host processor 102.

[0010] In the system 100 of FIG. 1a, the host processor 102 can be implemented using one or more processors, one or more multi-core processors, and/or any other suitable processor or processors. Further, each of the NVM devices 108.1-108.n can include nonvolatile memory such as NAND or NOR flash memory that uses a single bit per memory cell, multi-level cell (MLC) memory such as NAND flash memory with two bits per cell, polymer memory, phase-change memory (PCM), nanowire-based charge-trapping memory, ferroelectric transistor random access memory (FeTRAM), 3-dimensional cross-point memory, nonvolatile memory that uses memory resistor (memristor) technology, or any other suitable nonvolatile memory. In addition, the nonvolatile storage system 104 can be configured as a solid-state drive (SSD) or any other suitable persistent data storage medium, and the memory bus 105 may be implemented as a Peripheral Component Interconnect express (PCIe) bus, or any other suitable interface to the persistent data storage medium.

[0011] FIG. 1b depicts an exemplary functional view of the NVM read logic 112 included in the NVM controller 106. As shown in FIG. 1b, the NVM read logic 112 includes exclusive-or (XOR) logic 116, a multiplexer (MUX) 118, a hard data generator 119, a soft data generator 120, and an error correction decoder such as a low-density parity-check (LDPC) decoder 122, or any other suitable error correction decoder. In one embodiment, LDPC coding is employed to store data in at least some of the respective NVM devices 108.1-108.n in the form of LDPC encoded data (also referred to herein as "LDPC codewords"). For example, at least some of the NVM devices 108.1-108.n-1 may be used to store such LDPC codewords, and at least the NVM device 108.n may be used to store redundant data (also referred to herein as "XOR

data"), which may subsequently be employed to recover any decoded LDPC codewords found to be invalid or "faulty" due to, e.g., one or more error correction coding/decoding failures, etc. Further, such XOR data stored in the NVM device 108.n can be generated by performing exclusive-or (XOR) operations on the LDPC codewords stored in corresponding NVM storage locations of at least some of the respective NVM devices 108.1-108.n-1.

[0012] It is noted that nonvolatile memory storage can be organized within each of the NVM devices 108.1-108.n in a plurality of blocks of nonvolatile memory. Further, each block can include sixty-four (64) sequentially ordered pages, or any other suitable number of sequentially ordered pages. In addition, each page can include a plurality of NVM storage elements or cells for storing the LDPC encoded data or XOR data, as well as any metadata (e.g., valid/invalid markers, physical/logical addresses, a sequence number) that may have been generated for the page, or the data stored on the page.

[0013] In the system 100 of FIG. 1a, the NVM defect management policy engine 110 is operative to monitor a plurality of trigger events for detecting possible defects in one or more of the NVM devices 108.1-108.n included in the nonvolatile storage system 104, and to apply one or more defect management policies to the respective NVM devices 108.1-108.n based on the types of trigger events that resulted in detection of the possible defects. In one embodiment, such trigger events can include at least three (3) types of trigger events, namely, a hard bit read (HBR) trigger event, a soft bit read (SBR) trigger event, and an XOR data recovery trigger event. Further, each of the HBR trigger event, SBR trigger event, and XOR data recovery trigger event can be generated, from time to time, during the operation of reading data from the respective NVM devices 108.1-108.n using the NVM read logic 112.

[0014] The generation of HBR trigger events, SBR trigger events, and XOR data recovery trigger events will be further understood with reference to the following illustrative example, and FIGS. 1a-1b and 2. In this example, the NVM read logic 112 within the NVM controller 106 is employed to read data from one or more of the respective NVM devices 108.1-108.n in response to a data read request issued over the memory bus 105 by the host processor 102. For example, the host processor 102 may issue a data read request for reading a desired LDPC codeword stored at a specified NVM storage location within the NVM device 108.1. Further, the specified NVM storage location can include one or more NVM cells, each of which can be decoded to correspond to a 2-bit value, a 3-bit value, or any other suitable multi-bit value.

[0015] FIG. 2 depicts is a diagram of "voltage" versus "probability" that represents the decoding of the desired LDPC codeword stored at the specified NVM storage location within the NVM device 108.1. As illustrated in FIG. 2, an exemplary NVM cell included in the specified NVM storage location can, for example, be an MLC NAND cell with two bits per cell, which can be decoded to a 2-bit

value represented by one of a plurality of states (e.g., at least state 1 (11), state 2 (10), state 3 (00)). Specifically, the states 1, 2, 3 are illustrated with reference to three (3) curves 202, 204, 206, respectively, which represent the mathematical probabilities that the voltage along the horizontal axis corresponds to the various bit combinations (e.g., 11, 10, 00) that give rise to the respective states 1, 2, 3. Using the mathematical probabilities represented by the curves 202, 204, 206 of FIG. 2, what is referred to herein as a "hard bit read" or a "soft bit read" from the NVM cell can be decoded to correspond to a 2-bit value represented by one of the states 1, 2, 3.

[0016] When the NVM read logic 112 (see FIG. 1b) is to be used to perform a hard bit read from the exemplary NVM cell, at least one hard bit (HB) reference threshold voltage level can be established between adjacent ones of the states 1, 2, 3, ... (see FIG. 2). For example, at least a first HB reference threshold voltage level (HB1) can be established between the adjacent states 1, 2, at least a second HB reference threshold voltage level (HB2) can be established between the adjacent states 2, 3, and at least a third HB reference threshold voltage level (HB3) can be established between the state 3 and a next adjacent state (not shown). Further, a threshold voltage level of the NVM cell can be compared to at least these reference threshold voltage levels HB1, HB2, HB3 to determine the actual state 1, 2, 3, ..., of the NVM cell.

[0017] With reference to FIG. 2, if, when performing a hard bit read, it is determined that the reference threshold voltage level HB 1 is greater than the threshold voltage level of the exemplary NVM cell, then it can be concluded that the NVM cell is in the state 1. Further, if it is determined that the threshold voltage level of the NVM cell is greater than the reference threshold voltage level HB1 but less than the reference threshold voltage level HB2, then it can be concluded that the NVM cell is in the state 2. Moreover, if it is determined that the threshold voltage level of the NVM cell is greater than the reference threshold voltage level HB2 but less than the reference threshold voltage level HB3, then it can be concluded that the NVM cell is in the state 3.

[0018] When the NVM read logic 112 (see FIG. 1b) is to be used to perform a soft bit read from the exemplary NVM cell, at least one soft bit (SB) reference threshold voltage level can be established, e.g., along the horizontal axis within the regions of the curves 202, 204, 206 corresponding to the respective states 1, 2, 3, respectively, or at any other suitable location along the horizontal axis. For example, at least a first SB reference threshold voltage level (SB1) can be established within the region of the curve 204 corresponding to the state 2, and at least a second SB reference threshold voltage level (SB2) can be established within the region of the curve 206 corresponding to the state 3.

[0019] With reference to FIG. 2, if, when performing a soft bit read, it is determined that the threshold voltage level of the exemplary NVM cell is greater than the reference threshold voltage level HB1 but less than the ref-

erence threshold voltage level SB1, then it can be concluded that the threshold voltage level of the NVM cell is in the upper voltage region of the curve 202 corresponding to the state 1. Similarly, if it is determined that the threshold voltage level of the NVM cell is greater than the reference threshold voltage level HB2 but less than the reference threshold voltage level SB2, then it can be concluded that the threshold voltage level of the NVM cell is in the upper voltage region of the curve 204 corresponding to the state 2. In this way, such a soft bit read can allow the state of the NVM cell to be determined with increased resolution.

[0020] In response to the data read request issued by the host processor 102, the NVM read logic 112 (see FIGS. 1a, 1b) operates to control the MUX 118 to read the desired LDPC codeword from a selected one of the NVM devices 108.1-108.n-1. As discussed above, at least some of the NVM devices 108.1-108.n-1 may be used to store such LDPC codewords, and at least the NVM device 108.n may be used to store redundant data (XOR data).

[0021] In this example, the NVM read logic 112 controls the MUX 118 to first perform a hard bit read of the desired LDPC codeword from the selected NVM device. To that end, the MUX 118 receives data corresponding to the LDPC codeword from at least one NVM storage location of the selected NVM device, and provides the received data to the hard data generator 119, which uses the data to compare a threshold voltage level of at least one NVM cell included in the NVM storage location to one or more hard bit (HB) reference threshold voltage levels (e.g., HB1, HB2, HB3; see FIG. 2) (also referred to herein as the "center reference threshold voltage levels") to determine the state (e.g., state 1, 2, 3, ...; see FIG. 2) of the NVM cell. In one embodiment, the hard data generator 119 can further operate to compare the threshold voltage level of the NVM cell to one or more additional HB reference threshold voltage levels (e.g., HB1a, HB1b, HB2a, HB2b, HB3a, HB3b; see FIG. 2) offset from the respective center reference threshold voltage levels (e.g., HB1, HB2, HB3; see FIG. 2) to determine the state of the NVM cell. The hard data generator 119 then provides so-called "hard data" generated from the respective hard bit threshold voltage level comparisons to the LDPC decoder 122.

[0022] The NVM read logic 112 can further control the MUX 118 to perform a soft bit read of the LDPC codeword from the selected NVM device. To that end, the MUX 118 provides the data corresponding to the LDPC codeword to the soft data generator 120, which uses the data to compare the threshold voltage level of the NVM cell to one or more soft bit (SB) reference threshold voltage levels (e.g., SB1, SB2; see FIG. 2) to determine the state (e.g., state 1, 2, 3, ...; see FIG. 2) of the NVM cell. In one embodiment, the soft data generator 120 can further operate to compare the threshold voltage level of the NVM cell to one or more additional SB reference threshold voltage levels (e.g., SB1a, SB1b, SB2a, SB2b; see FIG. 2) offset from the respective SB1, SB2 reference thresh-

old voltage levels to determine the state of the NVM cell. The soft data generator 120 then provides so-called "soft data" generated from the respective soft bit threshold voltage level comparisons to the LDPC decoder 122.

[0023] Having received the hard data and the soft data from the hard data generator 119 and the soft data generator 120, respectively, the LDPC decoder 122 operates to first decode the LDPC codeword, using the hard data, to determine whether or not the LDPC codeword is valid. If the LDPC codeword is determined to be valid, then the LDPC decoder 122 forwards the decoded LDPC codeword to the host processor 102 as data output (see FIG. 1b). If, having been decoded using the hard data, the LDPC codeword is found to be invalid or "faulty" (e.g., there may have been an error correction coding/decoding failure), then the LDPC decoder 122 generates a trigger event(s) (see FIG. 1b), namely, an HBR trigger event, for receipt at the NVM defect management policy engine 110. It is noted that such a detection of a faulty LDPC codeword may indicate, among other things, that a physical memory page on the selected NVM device is being affected by an underlying hardware defect (e.g., an open or short circuit), or that the selected NVM device has an intrinsically high residual bit error rate (RBER).

[0024] Following one or more unsuccessful attempts to decode the LDPC codeword using the hard data, the LDPC decoder 122 further operates to decode the LDPC codeword using the soft data generated by the soft data generator 120. In one embodiment, the LDPC decoder 122 can implement an iterative probabilistic decoding process to decode the LDPC codeword by assigning initial probability metrics to each bit in the LDPC codeword. For example, such an iterative probabilistic decoding process can employ probability metrics in the form of logarithmic likelihood ratios (LLRs), in which a plurality of LLR values are provided as measures of the reliability that the bit values read from the selected NVM device are known. If the LDPC codeword is determined to be valid, then the LDPC decoder 122 can forward the decoded LDPC codeword to the host processor 102 as data output (see FIG. 1b). If, having been decoded using the soft data, the decoded LDPC codeword is again found to be faulty, then the LDPC decoder 122 generates a further trigger event(s) (see FIG. 1b), namely, an SBR trigger event, for receipt at the NVM defect management policy engine 110.

[0025] Following one or more unsuccessful attempts to decode the LDPC codeword using the soft data, the LDPC decoder 122 still further operates to perform XOR data recovery in an attempt to recover the desired LDPC codeword. To that end, the NVM read logic 112 (see FIGS. 1a, 1b) controls the XOR logic 116 to receive additional LDPC codewords stored in the currently unselected NVM devices 108.1-108.n-1, as well as XOR data stored in the NVM device 108.n. The XOR logic 116 operates to perform one or more exclusive-or (XOR) operations on these additional LDPC codewords and XOR data, and to provide the results of the XOR operations

as so-called "side data" to the LDPC decoder 122, which can perform additional LDPC decoding using at least these side data to recover the desired LDPC codeword.

[0026] Such additional LDPC decoding performed by the LDPC decoder 122 can involve any suitable XOR data recovery process. In one embodiment, the XOR data recovery process can include combining at least the additional LDPC codewords in the side data with the soft data, and performing LDPC decoding on the combined data. If at least some of the additional LDPC codewords fail to decode successfully, then one or more XOR operations can be performed on the successfully decoded and unsuccessfully decoded LDPC codewords, and the results of these XOR operations can undergo further LDPC decoding. If such further LDPC decoding is unsuccessful, then one or more additional XOR operations can be performed on the LDPC codewords, and the results of the additional XOR operations can be combined for still further LDPC decoding. If such further LDPC decoding is deemed to be successful, then it can be concluded that the desired LDPC codeword is recoverable. The LDPC decoder 122 can then provide the recovered LDPC codeword as data output for subsequent forwarding to the host processor 102. Otherwise, if such further LDPC decoding is again unsuccessful, then the LDPC decoder 122 generates an additional trigger event(s), namely, an XOR data recovery trigger event, for receipt at the NVM defect management policy engine 110.

[0027] As discussed above, the NVM defect management policy engine 110 is operative to monitor a plurality of trigger events for detecting possible defects in one or more of the NVM devices 108.1-108.n, and to apply one or more defect management policies to the respective NVM devices 108.1-108.n based on the types of trigger events that resulted in detection of the possible defects. The types of trigger events monitored by the NVM defect management policy engine 110 include an HBR trigger event, an SBR trigger event, and an XOR data recovery trigger event. Such defect management policies can be used proactively to retire memory in the nonvolatile storage system 104 with increased granularity, focusing the retirement of memory on regions of nonvolatile memory that are likely to contain a defect.

[0028] Such retirement of memory in the nonvolatile storage system 104 (see FIG. 1a) can involve the retirement (i.e., the discontinued use) of at least one block of nonvolatile memory, at least one page within at least one block of nonvolatile memory, at least one NVM storage location within at least one page, or any other suitable region of nonvolatile memory. For example, a physical memory block/page can be retired by relocating its data content to another physical memory block/page or other region of nonvolatile memory, and placing the retired physical memory block/page on one or more lists of blocks/pages that should not be subsequently allocated for storing data. A region of nonvolatile memory can also be retired by remapping an address for that region of nonvolatile memory to correspond to the address for an-

other region of nonvolatile memory. Such lists of retired blocks/pages and/or remapped addresses can be stored in volatile or nonvolatile memory associated with the NVM controller 106 (e.g., a memory 420b; see FIG. 4).

[0029] In one embodiment, in order to retire a region of nonvolatile memory in the nonvolatile storage system 104, the NVM defect management policy engine 110 can apply at least the following policies to the respective NVM devices 108.1-108.n:

Policy 1 - In response to an SBR trigger event and/or an XOR data recovery trigger event, (a) retire the NVM storage location of a codeword for which the host processor 102 issued a data read request, or (b) retire the physical memory page that includes the NVM storage location of the codeword for which the host processor 102 issued the data read request; and

Policy 2 - In response to a predetermined number of HBR trigger events, (a) retire the NVM storage location of a codeword for which the host processor 102 issued one or more data read requests, or (b) retire the physical memory page that includes the NVM storage location of the codeword for which the host processor 102 issued the data read requests.

[0030] It is noted that repeated application of Policy 1(a) above, in response to an XOR data recovery trigger event, may lead to an increased number of pages associated with SBR trigger events as the nonvolatile memory undergoes multiple program/erase (P/E) cycles. The application of Policy 1(a), in response to an SBR trigger event, may be more advantageous in maintaining a suitable bound on the SBR trigger event rate because each instantaneous SBR trigger event would be associated with a new page; any pages that may have been associated with prior SBR trigger events would have already been retired.

[0031] The application of Policy 1(b) above, in response to an SBR trigger event, may be considered to be more proactive than the application of Policy 1(a) because the intrinsic residual bit error rate (RBER) is generally uniform across a page. For example, if a codeword at one NVM storage location on a page causes an SBR trigger event, it is likely that another codeword at another NVM storage location on that page would cause another SBR trigger event. The application of Policy 1(b) may also reduce the memory requirements for storing lists of retired blocks/pages and/or remapped addresses.

[0032] The application of Policy 2(a) above, in response to a predetermined number of HBR trigger events, may be considered to be more proactive than the application of Policy 1(a), in response to an SBR trigger event. Further, the application of Policy 2(b) above, in response to a predetermined number of HBR trigger events, may be considered to be more proactive than the application of Policy 1(b), in response to an SBR trigger event. In general, as the respective policies for retiring

regions of nonvolatile memory become more proactive, more regions of the nonvolatile memory can be retired for the same P/E cycle count. Such proactive policies for retiring regions of nonvolatile memory can therefore lead to reduced over-provisioning, increased write amplification, and/or higher endurance.

[0033] An exemplary method of operating the NVM defect management policy engine 110 included in the nonvolatile storage system 104 is described below with reference to FIGS. 1a, 1b, and 3. As depicted in block 302 (see FIG. 3), the NVM defect management policy engine 110 (see FIG. 1a) monitors a plurality of trigger events for detecting possible defects in a selected one of the NVM devices 108.1-108.n (see FIGS. 1a, 1b). Such trigger events can include at least one HBR trigger event, at least one SBR trigger event, and/or at least one XOR data recovery trigger event. Further, such trigger events can be generated by the LDPC decoder 122 during one or more attempts to decode an LDPC codeword read from the selected NVM device. As depicted in block 304, a determination is made as to whether or not the type of trigger event corresponds to an SBR trigger event or an XOR data recovery trigger event. In the event the type of trigger event corresponds to the SBR trigger event or the XOR data recovery trigger event, at least one policy is applied, by the NVM defect management policy engine 110, including the policy of retiring the NVM storage location of the LDPC codeword for which the host processor 102 issued a data read request, or retiring the physical memory page that includes the NVM storage location of the LDPC codeword for which the host processor 102 issued the data read request, as depicted in block 306. As depicted in block 308, a further determination is made as to whether or not the type of trigger event corresponds to a predetermined number of HBR trigger events. In the event the type of trigger event corresponds to the predetermined number of HBR trigger events, at least one further policy is applied, by the NVM defect management policy engine 110, including the policy of retiring the NVM storage location of the LDPC codeword for which the host processor 102 issued one or more data read requests, or retiring the physical memory page that includes the NVM storage location of the LDPC codeword for which the host processor 102 issued the data read requests, as depicted in block 310.

[0034] FIG. 4 depicts an exemplary computer system 400 that can be configured to implement systems and methods of the claimed invention. As shown in FIG. 4, the computer system 400 can include at least one processor 402 communicably coupled to at least one memory 404 by a system bus 414, and communicably coupled to an NVM controller 420 by a memory bus 415. The computer system 400 can further include a keyboard 416 and a display 418 communicably coupled to the system bus 414, and an NVM defect management policy engine 430 and an NVM device 412 communicably coupled to the NVM controller 420. The NVM controller 420 includes at least one processor 420a operative to execute at least

one program out of at least one non-transitory storage medium, such as the memory 420b or any other suitable storage medium, to access persistent data storable in one or more blocks/pages within the NVM device 412, and to control operations of the NVM defect management policy engine 430. The processor 402 is operative to execute instructions stored on at least one non-transitory storage medium, such as the memory 404 or any other suitable storage medium, for performing various processes within the computer system 400, including one or more processes for controlling operations of the NVM controller 420. The memory 404 can include one or more memory components such as a volatile memory 410, which may be implemented as dynamic random access memory (DRAM) or any other suitable volatile memory. The memory 404 can also be configured to store an operating system 406 executable by the processor 402, as well as one or more applications 408 that may be run by the operating system 406. In response to a request generated by one of the applications 408, the processor 402 can execute the operating system 406 to perform desired data write/read operations on the volatile memory 410, and/or desired block/page write/read operations on the NVM device 412 via the NVM controller 420.

[0035] It is noted that FIG. 4 illustrates an exemplary embodiment of the computer system 400, and that other embodiments of the computer system 400 may include more components, or fewer components, than the components illustrated in FIG. 4. Further, the components may be arranged differently than as illustrated in FIG. 4. For example, in some embodiments, the NVM device 412 may be located at a remote site accessible to the computer system 400 via the Internet or any other suitable network. In addition, functions performed by various components contained in other embodiments of the computer system 400 may be distributed among the respective components differently than as described herein.

[0036] Having described the above exemplary embodiments of the disclosed systems and methods, other alternative embodiments or variations may be made. For example, it was described herein that a physical memory page with a defect(s) could be retired by relocating its data content to another physical memory page or other region of nonvolatile memory. Such data relocation generally means that the data on the defective physical memory page will be relocated to the other physical memory page as soon as possible to avoid further trigger events, such as SBR trigger events, occurring as a result of reading data from the defective page. After the data relocation is accomplished, the defective page (e.g., an upper page, a lower page) is to be retired permanently. In order to maintain a desired cell-to-cell coupling between neighboring word lines of the nonvolatile memory, as well as to maintain desired read-write buffer (RWB) margins on the neighboring word lines, a suitable random pattern of data can be programmed onto the retired page (upper or lower) each time a block/band containing the retired page is to be programmed. Further, if the retired page corre-

sponds to a lower page, then the associated upper page can be made more robust by modifying a pre-read voltage of the lower page during programming of the upper page. In this way, the modified pre-read voltage of the lower page can be used to effectively push potential errors from the upper page into the lower page, which is to be retired and never read again.

[0037] In addition, it was described herein that the application of Policy 1(b) above may reduce the memory requirements for storing lists of retired blocks/pages and/or remapped addresses. To further reduce such memory requirements, the granularity of memory retirement may be modified from a single page to a group of pages. For example, if such a group of pages includes N single-plane pages (e.g., for a dual plane page, N=2; for a quad plane page, N=4), then the memory requirements may be reduced by a factor of N. Moreover, the amount of stored information relating to the lists of retired blocks/pages and/or remapped addresses can be reduced to a simple bit map for all of the retired pages (or groups of pages), indicating whether or not those pages or groups of pages contain a defect(s).

[0038] Further modifications of the granularity of memory retirement are also possible. For example, in a random array of independent disks (RAID) scheme with bands configured as blocks across multiple NVM devices (e.g., dice), the same page can be retired for all blocks in the same stripe or die. Alternatively, the same page can be retired for all blocks if one or more HBR trigger events occur in more than a predetermined number of blocks for the same page in the same die.

[0039] Although illustrative examples of various embodiments of the disclosed subject matter are described herein, one of ordinary skill in the relevant art will appreciate that other manners of implementing the disclosed subject matter may alternatively be used. In the preceding description, various aspects of the disclosed subject matter have been described. For purposes of explanation, specific systems, apparatus, methods, and configurations were set forth in order to provide a thorough understanding of the disclosed subject matter. However, it will be apparent to one skilled in the relevant art having the benefit of this disclosure that the subject matter may be practiced without the specific details described herein. In other instances, well-known features, components, and/or modules were omitted, simplified, or combined in order not to obscure the disclosed subject matter.

[0040] It is noted that the term "operative to", as employed herein, means that a corresponding device, system, apparatus, etc., is able to operate, or is adapted to operate, for its desired functionality when the device, system, or apparatus is in its powered-on state. Moreover, various embodiments of the disclosed subject matter may be implemented in hardware, firmware, software, or some combination thereof, and may be described by reference to, or in conjunction with, program code such as instructions, functions, procedures, data structures, logic, application programs, design representations, and/or

formats for simulation, emulation, and/or fabrication of a design, which when accessed by a machine results in the machine performing tasks, defining abstract data types or low-level hardware contexts, or producing a result. It is further noted that the techniques illustrated in the drawing figures can be implemented using code and/or data stored and/or executed on one or more computing devices, such as general-purpose computers or computing devices. Such computers or computing devices store and communicate code and/or data (internally and/or with other computing devices over a network) using machine-readable media such as machine readable storage media {e.g., magnetic disks, optical disks, random access memory (RAM), read only memory (ROM), flash memory devices, phase-change memory) and machine readable communication media {e.g., electrical, optical, acoustical, or other form of propagated signals such as carrier waves, infrared signals, digital signals, etc.).

[0041] No element, operation, or instruction employed herein should be construed as critical or essential to the application unless explicitly described as such. Also, as employed herein, the article "a" is intended to include one or more items. Where only one item is intended, the term "one" or similar language is employed. Further, the phrase "based on" is intended to mean "based, at least in part, on" unless explicitly stated otherwise.

[0042] It is intended that the invention not be limited to the particular embodiments disclosed herein, but that the invention will include any and all particular embodiments and equivalents falling within the scope of the following appended claims.

Claims

1. A method of managing defects in a nonvolatile storage system (104), the nonvolatile storage system including one or more nonvolatile memory, NVM, devices (108.1-108.n), the method comprising:

receiving hard data from a hard data generator and soft data from a soft data generator;
decoding a codeword for which a host processor (102) issued a data read request,

- (i) wherein a first attempt to decode the codeword using the hard data is performed and, in response to decoding the codeword using the hard data and determining that the codeword is invalid, a hard bit read trigger event is generated,
- (ii) wherein, in response to unsuccessfully decoding the codeword using the hard data, a second attempt to decode the codeword using the soft data is performed and, in response to decoding the codeword using the soft data and determining that the codeword

is invalid, a soft bit read trigger event is generated, and

(iii) wherein, in response to unsuccessful decoding the codeword using the soft data, a third attempt to decode the codeword using exclusive-or, XOR, data generated by performing one or more XOR operations on codewords stored in corresponding locations of one or more NVM devices is performed and, in response to decoding the codeword using the XOR data and determining that the codeword is invalid, generating an XOR data recovery trigger event;

characterized by:

monitoring (302) a plurality of trigger events for detecting possible defects in at least one of the one or more NVM devices, each of the plurality of trigger events having an associated type;

having monitored the plurality of trigger events, determining types of the trigger events, wherein the types of the trigger events includes the soft bit read trigger event, the XOR data recovery trigger event, and a predetermined number of hard bit read trigger events; and

applying at least one defect management policy to the NVM device based on the monitored types of the trigger events, wherein the applying of the at least one defect management policy to the NVM device includes retiring an NVM storage location of the codeword for which the host processor (102) issued the data read request.

2. The method of claim 1, wherein the applying of the at least one defect management policy to the NVM device includes retiring a physical memory page that includes the NVM storage location of the codeword for which the host processor (102) issued the data read request.

3. The method of claim 2, further comprising: programming a region of nonvolatile memory containing the retired physical memory page comprising: programming a predetermined random pattern of data onto the retired physical memory page.

4. A nonvolatile storage system (104), comprising:

a nonvolatile memory, NVM, controller (106);
one or more NVM devices (108.1-108.n) communicably coupled to the NVM controller; and
an error correction decoder (122) of the NVM controller, wherein the error correction decoder is operative to:

receive hard data from a hard data generator and soft data from a soft data generator; and
 decode a codeword for which a host processor (102) issued a data read request, (i) wherein a first attempt to decode the codeword using the hard data is performed and, in response to decoding the codeword using the hard data and determining that the codeword is invalid, a hard bit read trigger event is generated, (ii) wherein, in response to unsuccessfully decoding the codeword using the hard data, a second attempt to decode the codeword using the soft data is performed and, in response to decoding the codeword using the soft data and determining that the codeword is invalid, a soft bit read trigger event is generated, and (iii) wherein, in response to unsuccessful decoding the codeword using the soft data, a third attempt to decode the codeword using exclusive-or, XOR, data generated by performing one or more XOR operations on codewords stored in corresponding locations of one or more NVM devices is performed and, in response to decoding the codeword using the XOR data and determining that the codeword is invalid, generating an XOR data recovery trigger event; **characterized in that** the nonvolatile storage system further comprises means (110) for :

monitoring a plurality of trigger events for detecting possible defects in at least one of the one or more NVM devices, each of the plurality of trigger events having an associated type;
 having monitored the plurality of trigger events, determining types of the trigger events, wherein the types of the trigger events include the soft bit read trigger event, the XOR data recovery trigger event, and a predetermined number of hard bit read trigger events; and
 applying at least one defect management policy to the NVM device based on the monitored types of the trigger events, wherein the applying of the at least one defect management policy to the NVM device includes retiring an NVM storage location of the codeword for which the host processor (102) issued the data read request.

5. The system of claim 4, wherein the nonvolatile storage system, having monitored the soft bit read trigger event, the XOR data recovery trigger event and

the predetermined number of hard bit read trigger events, further comprises means (110) for retiring a physical memory page that includes the NVM storage location of the codeword for which the host processor (102) issued the data read request.

6. Machine-readable storage including machine-readable instructions executable by a processor that, when executed, cause the processor to implement a method as claimed in any of claims 1-3.

Patentansprüche

1. Verfahren zum Verwalten von Defekten in einem nichtflüchtigen Speichersystem (104), wobei das nichtflüchtige Speichersystem ein oder mehrere NVM-(Nonvolatile Memory)-Vorrichtungen (108.1-108.n) enthält, wobei das Verfahren Folgendes beinhaltet:

Empfangen harter Daten von einem Harte-Daten-Generator und weicher Daten von einem Weiche-Daten-Generator;

Decodieren eines Codeworts, für das ein Host-Prozessor (102) eine Datenleseanforderung ausgegeben hat,

(i) wobei ein erster Versuch zum Decodieren des Codeworts anhand der harten Daten durchgeführt und als Reaktion auf das Decodieren des Codeworts anhand der harten Daten und das Feststellen, dass das Codewort ungültig ist, ein Hard-Bit-Lesetriggerereignis erzeugt wird,

(ii) wobei als Reaktion auf ein erfolgloses Decodieren des Codeworts anhand der harten Daten ein zweiter Versuch zum Decodieren des Codeworts anhand der weichen Daten durchgeführt wird und als Reaktion auf das Decodieren des Codeworts anhand der weichen Daten und das Feststellen, dass das Codewort ungültig ist, ein Soft-Bit-Lesetriggerereignis erzeugt wird, und

(iii) wobei als Reaktion auf ein erfolgloses Decodieren des Codeworts anhand der weichen Daten ein dritter Versuch zum Decodieren des Codeworts anhand von XOR-(Exklusive-Or)-Daten, erzeugt durch Ausführen einer oder mehrerer XOR-Operationen an an entsprechenden Stellen einer oder mehrerer NVM-Vorrichtungen gespeicherten Codewörtern, durchgeführt wird, und als Reaktion auf das Decodieren des Codeworts anhand der XOR-Daten und das Feststellen, dass das Codewort ungültig ist, ein XOR-Datenwiederherstellungs-Triggerereignis erzeugt wird;

gekennzeichnet durch:

- Überwachen (302) mehrerer Triggerereignisse zum Erfassen möglicher Defekte in mindestens einer der ein oder mehreren NVM-Vorrichtungen, wobei mit jedem der mehreren Triggerereignisse ein Typ assoziiert ist; 5
- Bestimmen, nach dem Überwachen der mehreren Triggerereignisse, von Typen der Triggerereignisse, wobei die Typen der Triggerereignisse das Soft-Bit-Lesetriggerereignis, das XOR-Datenwiederherstellungs-Triggerereignis und eine vorbestimmte Anzahl von Hard-Bit-Lesetriggerereignissen umfassen; und 10
- Anwenden mindestens einer Defektverwaltungsrichtlinie auf die NVM-Vorrichtung auf der Basis der überwachten Typen der Triggerereignisse, wobei das Anwenden der mindestens einen Defektverwaltungsrichtlinie auf die NVM-Vorrichtung das Zurückziehen einer NVM-Speicherstelle des Codeworts beinhaltet, für das der Host-Prozessor (102) die Datenleseanforderung ausgegeben hat. 15 20 25
2. Verfahren nach Anspruch 1, wobei das Anwenden der mindestens einen Defektverwaltungsrichtlinie auf die NVM-Vorrichtung das Zurückziehen einer physischen Speicherseite beinhaltet, die die NVM-Speicherstelle des Codeworts enthält, für das der Host-Prozessor (102) die Datenleseanforderung ausgegeben hat. 30 35
3. Verfahren nach Anspruch 2, das ferner Folgendes beinhaltet:
 Programmieren einer Region von nichtflüchtigem Speicher, die die zurückgezogene physische Speicherseite enthält, das Folgendes beinhaltet: Programmieren eines vorbestimmten Zufallsmusters von Daten auf die zurückgezogene physische Speicherseite. 40
4. Nichtflüchtiges Speichersystem (104), das Folgendes umfasst: 45
- einen NVM-(Nonvolatile Memory)-Controller (106);
 eine oder mehrere NVM-Vorrichtungen (108.1-108.n), die kommunikativ mit dem NVM-Controller gekoppelt sind; und
 einen Fehlerkorrektur-Decoder (122) des NVM-Controllers, wobei der Fehlerkorrektur-Decoder dient zum: 50 55
- Empfangen harter Daten von einem Harte-Daten-Generator und weicher Daten von ei-

nem Weiche-Daten-Generator;

Decodieren eines Codeworts, für das ein Host-Prozessor (102) eine Datenleseanforderung ausgegeben hat, (i) wobei ein erster Versuch zum Decodieren des Codeworts anhand der harten Daten durchgeführt und als Reaktion auf das Decodieren des Codeworts anhand der harten Daten und das Feststellen, dass das Codewort ungültig ist, ein Hard-Bit-Lesetriggerereignis erzeugt wird, (ii) wobei als Reaktion auf ein erfolgloses Decodieren des Codeworts anhand der harten Daten ein zweiter Versuch zum Decodieren des Codeworts anhand der weichen Daten durchgeführt wird und als Reaktion auf das Decodieren des Codeworts anhand der weichen Daten und das Feststellen, dass das Codewort ungültig ist, ein Soft-Bit-Lesetriggerereignis erzeugt wird, und (iii) wobei als Reaktion auf ein erfolgloses Decodieren des Codeworts anhand der weichen Daten ein dritter Versuch zum Decodieren des Codeworts anhand von XOR-(Exklusive-or)-Daten, erzeugt durch Ausführen einer oder mehrerer XOR-Operationen an an entsprechenden Stellen einer oder mehrerer NVM-Vorrichtungen gespeicherten Codewörtern, durchgeführt wird, und als Reaktion auf das Decodieren des Codeworts anhand der XOR-Daten und das Feststellen, dass das Codewort ungültig ist, ein XOR-Datenwiederherstellungs-Triggerereignis erzeugt wird;

dadurch gekennzeichnet, dass das nichtflüchtige Speichersystem ferner Mittel (110) umfasst zum:

Überwachen mehrerer Triggerereignisse zum Erfassen möglicher Defekte in mindestens einer der ein oder mehreren NVM-Vorrichtungen, wobei mit jedem der mehreren Triggerereignisse ein Typ assoziiert ist;
 Bestimmen, nach dem Überwachen der mehreren Triggerereignisse, von Typen der Triggerereignisse, wobei die Typen der Triggerereignisse das Soft-Bit-Lesetriggerereignis, das XOR-Datenwiederherstellungs-Triggerereignis und eine vorbestimmte Anzahl von Hard-Bit-Lesetriggerereignissen umfassen; und
 Anwenden mindestens einer Defektverwaltungsrichtlinie auf die NVM-Vorrichtung auf der Basis der überwachten Typen der Triggerereignisse, wobei das Anwenden der mindestens einen Defektverwaltungsrichtlinie auf die NVM-Vorrichtung das Zurückziehen einer NVM-Speicherstelle des Co-

deworts beinhaltet, für das der Host-Prozessor (102) die Datenleseanforderung ausgegeben hat.

5. System nach Anspruch 4, wobei das nichtflüchtige Speichersystem, das das Soft-Bit-Lesetriggerereignis, das XOR-Datenwiederherstellungs-Triggerereignis und die vorbestimmte Anzahl von Hard-Bit-Lesetriggerereignissen überwacht hat, ferner Mittel (110) zum Zurückziehen einer physischen Speicherseite umfasst, die die NVM-Speicherstelle des Codeworts enthält, für das der Host-Prozessor (102) die Datenleseanforderung ausgegeben hat. 5 10
6. Maschinenlesbarer Speicher, der maschinenlesbare Befehle enthält, die von einem Prozessor ausgeführt werden können, die bei Ausführung bewirken, dass der Prozessor ein Verfahren nach einem der Ansprüche 1-3 implementiert. 15 20

Revendications

1. Un procédé de gestion de défauts dans un système à espace mémoire non volatile (104), le système à espace mémoire non volatile comprenant un ou plusieurs dispositifs à mémoire non volatile, NVM, (108.1-108.n), le procédé comprenant : 25 30
la réception de données dures à partir d'un générateur de données dures et de données molles à partir d'un générateur de données molles, le décodage d'un mot de code pour lequel un processeur hôte (102) a émis une demande de lecture de données, 35
(i) où une première tentative de décodage du mot de code au moyen des données dures est exécutée et, en réponse au décodage du mot de code au moyen des données dures et à la détermination que le mot de code est non valide, un événement de déclenchement de lecture de bits durs est généré, 40
(ii) où, en réponse à un décodage infructueux du mot de code au moyen des données dures, une deuxième tentative de décodage du mot de code au moyen des données molles est exécutée et, en réponse au décodage du mot de code au moyen des données molles et à la détermination que le mot de code est non valide, un événement de déclenchement de lecture de bits mous est généré, et 50
(iii) où, en réponse à un décodage infructueux du mot de code au moyen des données molles, une troisième tentative de décodage du mot de code au moyen de don- 55

nées OU exclusif, XOR, générées par l'exécution d'une ou de plusieurs opérations XOR sur des mots de code conservés en mémoire dans des emplacements correspondants d'un ou de plusieurs dispositifs NVM est exécutée et, en réponse au décodage du mot de code au moyen des données XOR et à la détermination que le mot de code est non valide, la génération d'un événement de déclenchement de récupération de données XOR,

caractérisé par :

la surveillance (302) d'une pluralité d'événements de déclenchement en vue de la détection de défauts possibles dans au moins un des un ou plusieurs dispositifs NVM, chaque événement de déclenchement de la pluralité d'événements de déclenchement possédant un type associé, après la surveillance de la pluralité d'événements de déclenchement, la détermination de types des événements de déclenchement, où les types des événements de déclenchement comprennent l'événement de déclenchement de lecture de bits mous, l'événement de déclenchement de récupération de données XOR et un nombre prédéterminé d'événements de déclenchement de lecture de bits durs, et l'application d'au moins une politique de gestion de défauts au dispositif NVM en fonction des types surveillés des événements de déclenchement, où l'application de la au moins une politique de gestion de défauts au dispositif NVM comprend le retrait d'un emplacement d'espace mémoire NVM du mot de code pour lequel le processeur hôte (102) a émis la demande de lecture de données.

2. Le procédé selon la Revendication 1, où l'application de la au moins une politique de gestion de défauts au dispositif NVM comprend le retrait d'une page de mémoire physique qui comprend l'emplacement d'espace mémoire NVM du mot de code pour lequel le processeur hôte (102) a émis la demande de lecture de données.
3. Le procédé selon la Revendication 2, comprenant en outre : la programmation d'une zone de mémoire non volatile contenant la page de mémoire physique retirée comprenant : la programmation d'un modèle aléatoire prédéterminé de données vers la page de mémoire physique retirée.

4. Un système à espace mémoire non volatile (104), comprenant :

un dispositif de commande de mémoire non volatile, NVM, (106),
un ou plusieurs dispositifs NVM (108.1-108.n) couplés en communication au dispositif de commande NVM, et
un décodeur de correction d'erreurs (122) du dispositif de commande NVM, où le décodeur de correction d'erreurs est conçu de façon à :

recevoir des données dures à partir d'un générateur de données dures et des données molles à partir d'un générateur de données molles, et
décoder un mot de code pour lequel un processeur hôte (102) a émis une demande de lecture de données, (i) où une première tentative de décodage du mot de code au moyen des données dures est exécutée et, en réponse au décodage du mot de code au moyen des données dures et à la détermination que le mot de code est non valide, un événement de déclenchement de lecture de bits durs est généré, (ii) où, en réponse à un décodage infructueux du mot de code au moyen des données dures, une deuxième tentative de décodage du mot de code au moyen des données molles est exécutée et, en réponse au décodage du mot de code au moyen des données molles et à la détermination que le mot de code est non valide, un événement de déclenchement de lecture de bits mous est généré, et (iii) où, en réponse à un décodage infructueux du mot de code au moyen des données molles, une troisième tentative de décodage du mot de code au moyen de données OU exclusif, XOR, générées par l'exécution d'une ou de plusieurs opérations XOR sur des mots de code conservés en mémoire dans des emplacements correspondants d'un ou de plusieurs dispositifs NVM est exécutée et, en réponse au décodage du mot de code au moyen des données XOR et à la détermination que le mot de code est non valide, la génération d'un événement de déclenchement de récupération de données XOR,
caractérisé en ce que le système à espace mémoire non volatile comprend en outre un moyen (110) destiné à :

la surveillance d'une pluralité d'événements de déclenchement en vue de la détection de défauts possibles dans au moins un dispositif NVM des un ou plusieurs dispositifs NVM, chaque événe-

ment de déclenchement de la pluralité d'événements de déclenchement possédant un type associé, après la surveillance de la pluralité d'événements de déclenchement, la détermination de types des événements de déclenchement, où les types des événements de déclenchement comprennent l'événement de déclenchement de lecture de bits mous, l'événement de déclenchement de récupération de données XOR et un nombre prédéterminé d'événements de déclenchement de lecture de bits durs, et l'application d'au moins une politique de gestion de défauts au dispositif NVM en fonction des types surveillés des événements de déclenchement, où l'application de la au moins une politique de gestion de défauts au dispositif NVM comprend le retrait d'un emplacement d'espace mémoire NVM du mot de code pour lequel le processeur hôte (102) a émis la demande de lecture de données.

5. Le système selon la Revendication 4, où le système à espace mémoire non volatile, après la surveillance de l'événement de déclenchement de lecture de bits mous, l'événement de déclenchement de récupération de données XOR et le nombre prédéterminé d'événements de déclenchement de lecture de bits durs, comprend en outre un moyen (110) de retrait d'une page de mémoire physique qui comprend l'emplacement d'espace mémoire NVM du mot de code pour lequel le processeur hôte (102) a émis la demande de lecture de données.
6. Un espace mémoire lisible par ordinateur contenant des instructions lisibles par ordinateur exécutables par un processeur qui, lorsqu'elles sont exécutées, amènent le processeur à mettre en oeuvre un procédé selon l'une quelconque des Revendications 1 à 3.

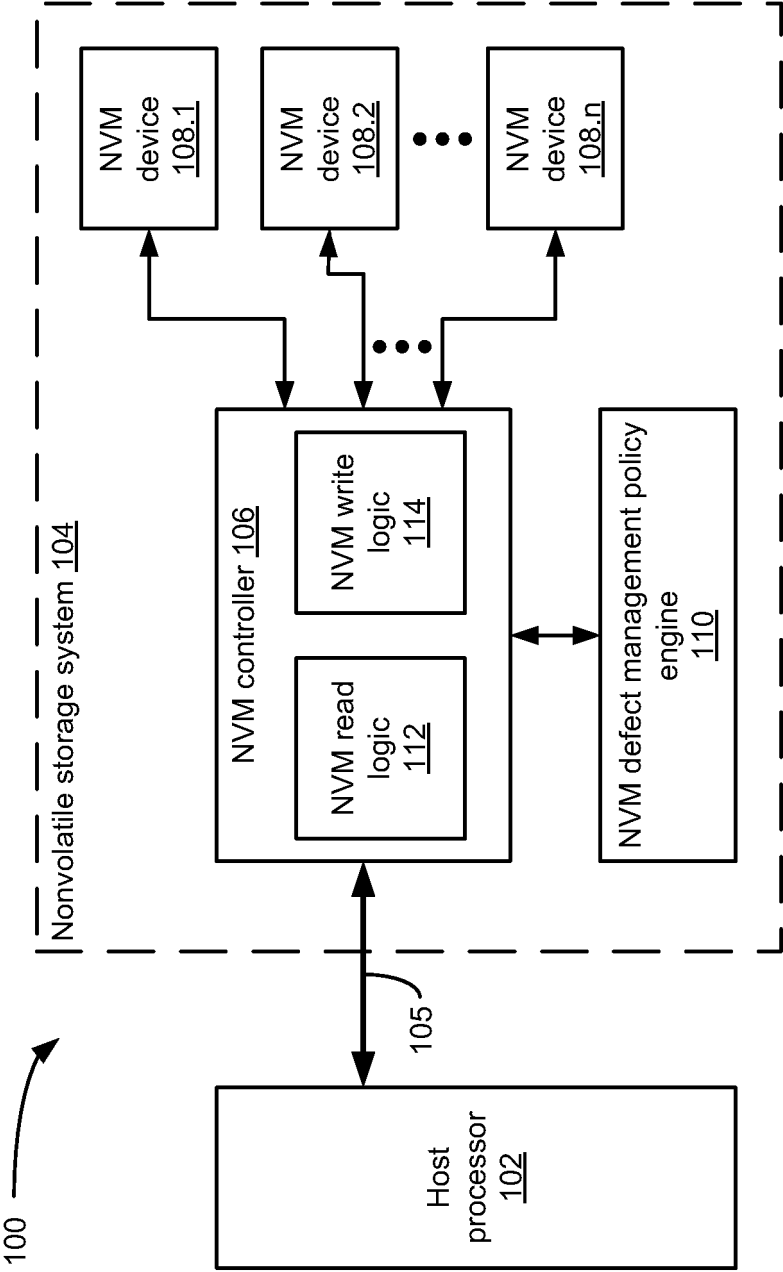


FIG. 1a

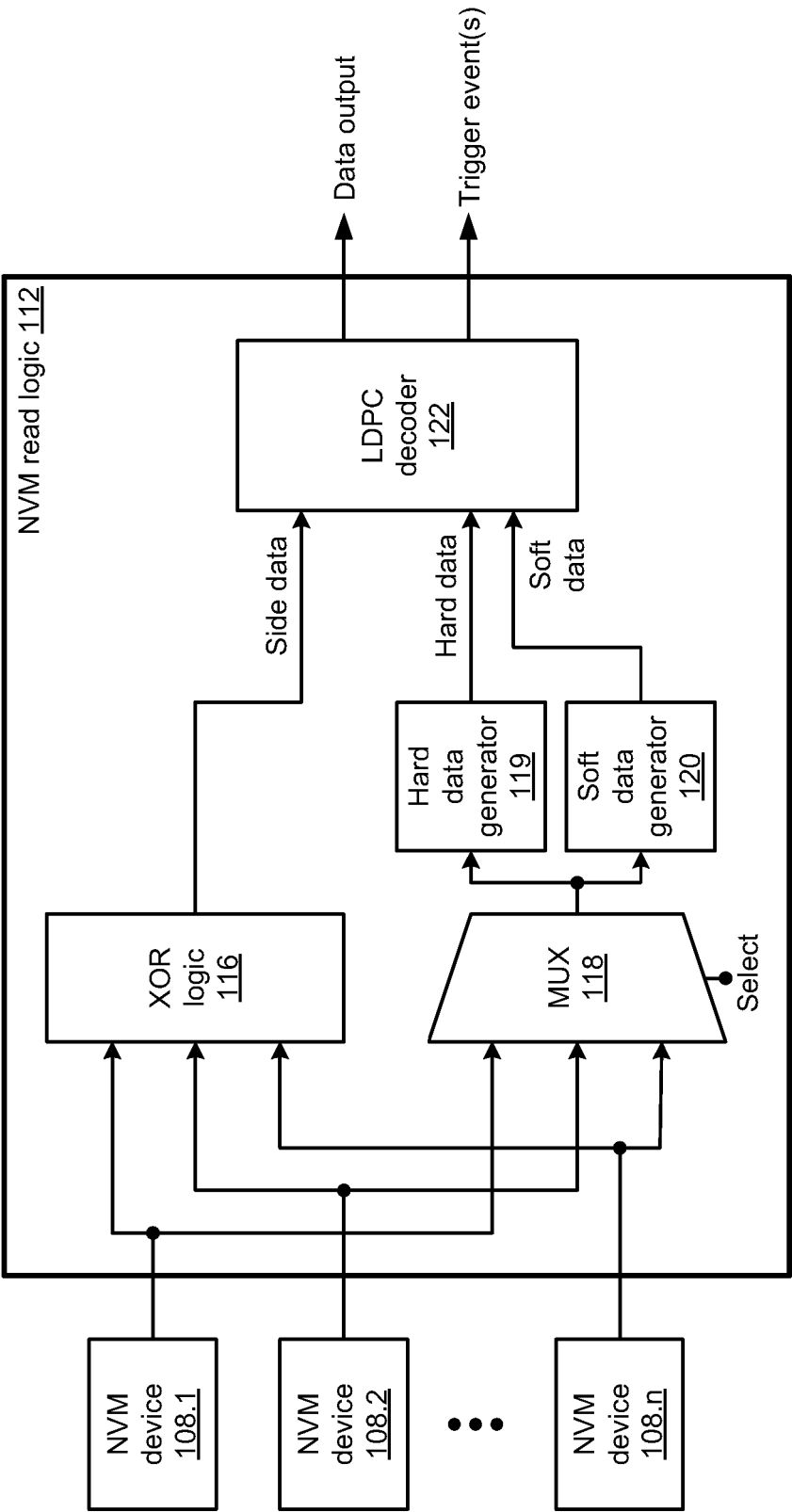


FIG. 1b

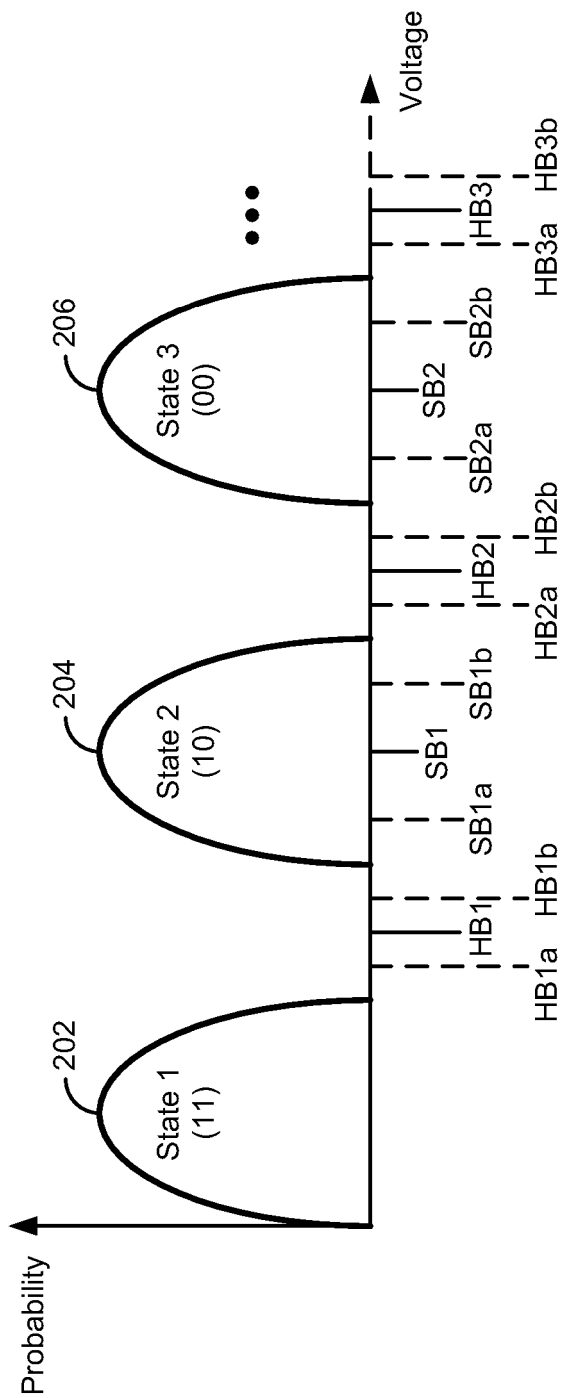
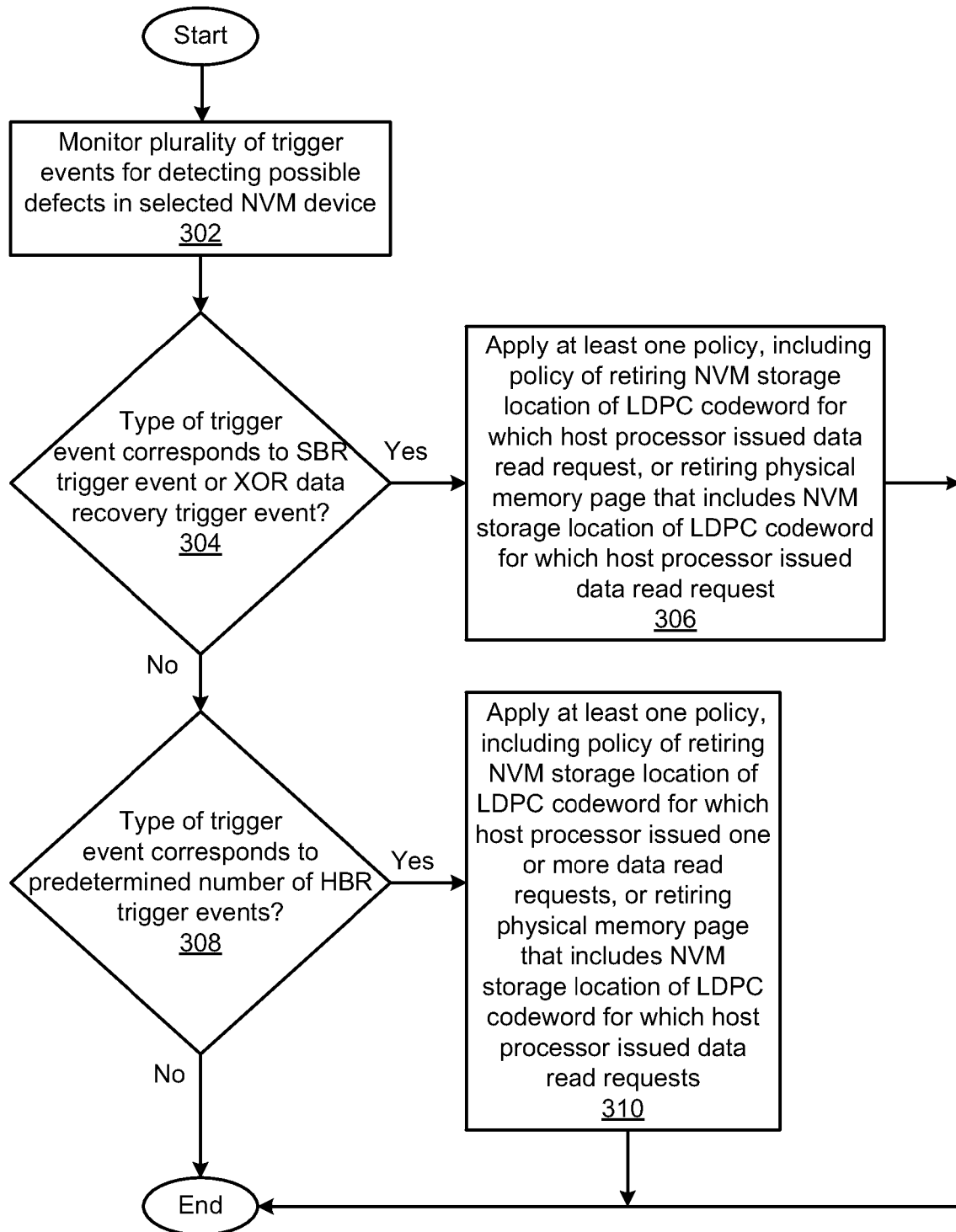
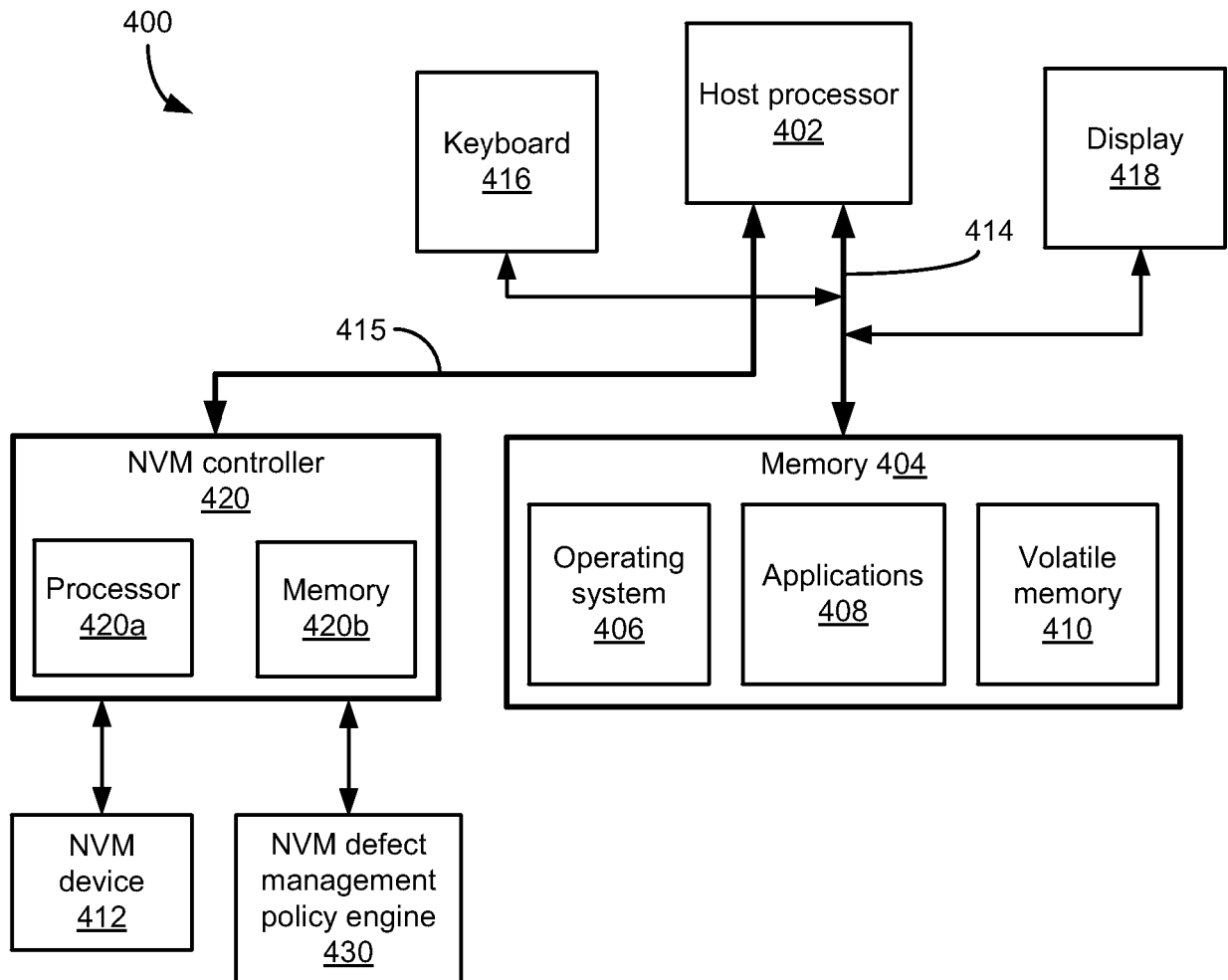


FIG. 2

**FIG. 3**

**Fig. 4**

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- US 2012159285 A1 [0005]